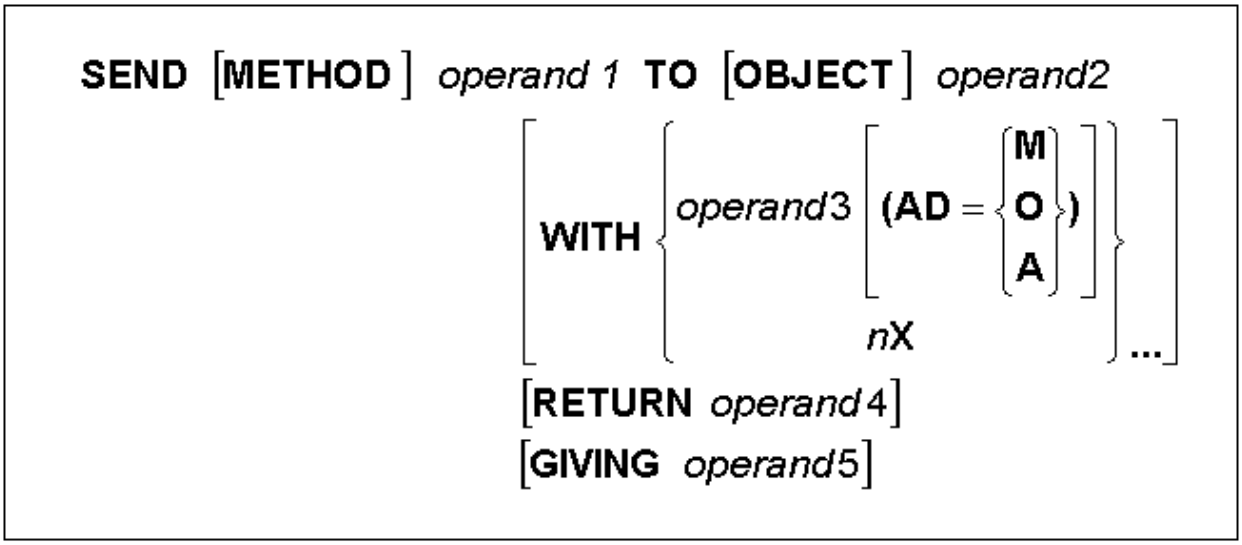
SEND METHOD



Operand	Possible Structure					Possible Formats											Referencing Permitted	Dynamic Definition
Operand1	C	S				A											yes	no
Operand2		S														O	no	no
Operand3	C	S	A	G		A	N	P	I	F	B	D	T	L	C	G	O	yes	no
Operand4		S	A			A	N	P	I	F	B	D	T	L	C	G	O	yes	no
Operand5		S			N				I									yes	no

The formats C and G can only be passed to methods of local classes. For more information, see the section Local Classes.

Function

The SEND METHOD statement is used to invoke a particular method of an object.

Note: Optional parameters (*n*X notation) are available with Version 4.1.1 and all subsequent releases. The AD option is available with Version 4.1.2 and all subsequent releases.

Method-Name - operand1

Operand1 is the name of a method which is supported by the object specified in *operand2*.

Since the method names can be identical in different interfaces of a class, the method name in *operand1* can also be qualified with the interface name to avoid ambiguity.

In the following example, the object #O3 has an interface Iterate with the method Start. The following statements apply:

```
* Specifying only the method name.
  SEND "Start" TO #O3
  * Qualifying the method name with the interface name.
  SEND "Iterate.Start" TO #O3
```

If no interface name is specified, Natural searches the method name in all the interfaces of the class. If the method name is found in more than one interface, a runtime error occurs.

Object Handle - operand2

The handle of the object to which the method call is to be sent.

Operand2 must be defined as an object handle (HANDLE OF OBJECT). The object must already exist.

To invoke a method of the current object inside a method, use the system variable *THIS-OBJECT.

Parameter - operand3

As *operand3* you can specify parameters specific to the method.

In the following example, the object #O3 has the method PositionTo with the parameter Pos. The method is called in the following way:

```
  SEND "PositionTo" TO #O3 WITH Pos
```

Methods can have optional parameters. Optional parameters need not to be specified when the method is called. To omit an optional parameter, use the placeholder 1X. To omit n optional parameters, use the placeholder *n*X.

In the following example, the method SetAddress of the object #O4 has the parameters FirstName, MiddleInitial, LastName, Street and City, where MiddleInitial, Street and City are optional. The following statements apply:

```
* Specifying all parameters.
  SEND "SetAddress" TO #O4 WITH FirstName MiddleInitial LastName Street City
  * Omitting one optional parameter.
  SEND "SetAddress" TO #O4 WITH FirstName 1X LastName Street City
  * Omitting all optional parameters.
  SEND "SetAddress" TO #O4 WITH FirstName 1X LastName 2X
```

Omitting a non-optional (mandatory) parameter results in a runtime error.

AD=

If *operand3* is a variable, you can mark it in one of the following ways:

AD=O	Non-modifiable, see Session Parameter AD=O.
AD=M	Modifiable, see Session Parameter AD=M. This is the default setting.
AD=A	Input only, see Session Parameter AD=A.

If *operand3* is a constant, AD cannot be explicitly specified. For constants AD=O always applies.

Parameter - nX

With the notation nX you can specify that the next n parameters are to be skipped (for example, 1X to skip the next parameter, or 3X to skip the next three parameters). This means that for the next n parameters no values are passed to the method.

For a method implemented in Natural, a parameter that is to be skipped must be defined with the keyword OPTIONAL in the method subprogram's DEFINE DATA PARAMETER statement. OPTIONAL means that a value can - but need not - be passed from the invoking object to such a parameter.

RETURN - operand4

If the RETURN clause is omitted and the method has a return value, the return value is discarded.

If the RETURN clause is specified, *operand4* contains the return value of the method. If the method execution fails, *operand4* is reset to its initial value.

Note:
For classes written in Natural, the return value of a method is defined by entering one additional parameter in the parameter data area of the method and by marking it with 'BY VALUE RESULT'. (For more information, see the section PARAMETER Clause.) Therefore the parameter data area of a method that is written in Natural and that has a return value always contains one additional field next to the method parameters. This is to be considered when you call a method of a Natural written class and want to use the parameter data area of the method in the SEND statement.

GIVING - operand5

If the GIVING clause is not specified, the Natural run time error processing is triggered if an error occurs.

If the GIVING clause is specified, *operand5* contains the Natural message number if an error occurred, or zero on success.

